Face Flashing: a Secure Liveness Detection Protocol based on Light Reflections

Di Tang
Chinese University of Hong Kong
td016@ie.cuhk.edu.hk

Zhe Zhou
Fudan University
zhouzhe@fudan.edu.cn

Yinqian Zhang
Ohio State University
yinqian@cse.ohio-state.edu

Kehuan Zhang
Chinese University of Hong Kong
khzhang@ie.cuhk.edu.hk

***Abstract*—Face authentication systems are becoming increasingly prevalent, especially with the rapid development of Deep Learning technologies. However, human facial information is easy to be captured and reproduced, which makes face authentication systems vulnerable to various attacks. *Liveness detection* is an important defense technique to prevent such attacks, but existing solutions did not provide clear and strong security guarantees, especially in terms of time.**

To overcome these limitations, we propose a new liveness detection protocol called *Face Flashing* that significantly increases the bar for launching successful attacks on face authentication systems. By randomly flashing well-designed pictures on a screen and analyzing the reflected light, our protocol has leveraged physical characteristics of human faces: reflection processing at the speed of light, unique textual features, and uneven 3D shapes. Cooperating with working mechanism of the screen and digital cameras, our protocol is able to detect subtle traces left by an attacking process.

To demonstrate the effectiveness of Face Flashing, we implemented a prototype and performed thorough evaluations with large data set collected from real-world scenarios. The results show that our Timing Verification can effectively detect the time gap between legitimate authentications and malicious cases. Our Face Verification can also differentiate 2D plane from 3D objects accurately. The overall accuracy of our liveness detection system is 98.8%, and its robustness was evaluated in different scenarios. In the worst case, our system's accuracy decreased to a still-high 97.3%.

I. Introduction

User authentication is a fundamental security mechanism. However, passwords, the most widely used certificate for authentication, have widely known drawbacks in security and usability: strong passwords are difficult to memorize, whereas convenient ones provide only weak protection. Therefore, researchers have long sought alternative security certificates and methods, among which biometric authentication is a promising candidate. Biometric authentication verifies inherent factors instead of knowledge factors (e.g., passwords) and possession factors (e.g., secure tokens). Some biometric-based schemes have already been proposed. They exploit users' fingerprints, voice spectra, and irises. Face-based schemes have become increasingly widespread because of rapid developments in face recognition technologies and deep learning algorithms.

However, in contrast to other biometrics (i.e., iris and retina recognition) that are difficult for adversaries to acquire and duplicate, human faces can be easily captured and reproduced, which makes face authentication systems vulnerable to attacks. For example, adversaries could obtain numerous photographs and facial videos from social networks or stolen smartphones. Furthermore, these images can be easily utilized to build facial models of target individuals and bypass face authentication systems, benefiting from architectural advances in General-Purpose Graphics Processing Unit (GPGPU) and advanced image synthesizing techniques. Such attacks can be as simple as presenting a printed photograph, or as sophisticated as dynamically generating video streams by using video morphing techniques.

To counter such attacks, *liveness detection* methods have been developed during the past decade. Crucial to such methods are challenge-response protocols, in which challenges are sent to the user who then responds in accordance with displayed instructions. The responses are subsequently captured and verified to ensure that they come from a real human being instead of being synthesized. Challenges typically adopted in studies have included blinking, reading words or numbers aloud, head movements, and handheld camera movements.

However, these methods do not provide a strong security guarantee. Adversaries may be able to bypass them by using modern computers and technology. More specifically, as Li et al. [17] argued, many existing methods are vulnerable to media-based facial forgery (MFF) attacks. Adversaries have even been able to bypass FaceLive, the method proposed by Li et al. and designed to defend against MFF attacks, by deliberately simulating the authentication environment.

We determined that the root cause of this vulnerability is the lack of strict time verification of a response. That is, the time required for a human to respond to a movement challenge is long and varies among individuals. Adversaries can synthesize responses faster than legitimate users by using modern hardware and advanced algorithms. Therefore, previous protocols could not detect liveness solely on the basis of response time.

To address this vulnerability, we propose a new challengeresponse protocol called Face Flashing. The core proposal of this protocol is to emit light of random colors from a liquid-

Network and Distributed Systems Security (NDSS) Symposium 2018
18-21 February 2018, San Diego, CA, USA
ISBN 1-891562-49-5
http://dx.doi.org/10.14722/ndss.2018.23176
www.ndss-symposium.org

crystal display (LCD) screen (our challenge) and use a camera to capture the light reflected from the face (our response). The response generation process requires negligible time, whereas forging the response would require substantially more time. By leveraging this substantial difference, Face Flashing thus provides effective security in terms of time.

The security of the Face Flashing protocol is based on two factors: time and shape. We use linear regression models and a neural network model to verify each factor, respectively. Our verification of time ensures that the response has not been falsified, whereas verification of shape ensures that the face shape is stereo and face-like. By using these two verifications, our protocol simultaneously satisfies the three essentials of a secure liveness detection protocol. First, we leverage an unpredictable challenge, flashing a sequence of effectively designed, randomly generated images. Second, our responses are difficult to forge not only because of the difference in time but also in the effort required to generate responses. In particular, legitimate users need not perform any extra steps, and legitimate responses are generated automatically (through light reflection) and instantaneously, whereas adversaries must expend substantially more effort to synthesize quality responses to bypass our system. Third, we can effectively verify the genuineness of responses by using our challenges. Specifically, we verify users on the basis of the received responses; for example, by checking whether the shiny area in the response accords with the challenge (lighting area in challenges will always produce highly intensive responses in a local area). The detailed security analysis and our adversary model are presented in later sections of this paper.

Contributions. Our paper's contributions are three-fold:

- **A new liveness detection protocol, Face Flashing.** We propose Face Flashing, a new liveness detection protocol, that flashes randomly generated colors and verifies the reflected light. In our system, adversaries do not have the time required to forge responses during authentication.
- **Effective and efficient verifications of timing and face.** By employing working mechanisms of screens and digital cameras, we design a method that uses linear regression models to verify the time. Furthermore, by using a well-designed neural network model, our method verifies the face shape. By combining these two verification procedures, our protocol provides strong security.
- **Implementation of a prototype and evaluations.** We implement a prototype and conduct thorough evaluations. The evaluation results suggest that our method performs reliably in different settings and is highly accurate.

Roadmap. This paper is organized as follows: Section II introduces the background and Section III describes our adversary model and preset assumptions. Section IV details the design of our protocol. Section V presents the security analysis. Section VI elucidates experiment settings and evaluation results. Section VII summarizes related works. Section VIII and Section IX discusses limitations and future works. Finally, Section X concludes this paper.

II. Background

In this section, we describe the typical architecture of face-based authentication systems (Section II-A). Subsequently, we briefly review attacks and solutions of liveness detection (Section II-B).

A. Architecture of Face Authentication Systems

A typical architecture of face authentication system is illustrated in Fig 1. It is divided into two parts: front-end devices and the back-end server. The front-end devices comprises camera and auxiliary sensors such as flash lamps, microphones. The back-end server contains two main modules: a liveness detection module and a face recognition module. When the user commences the authentication process, the liveness detection module is initiated and sends generated parameters to front-end devices (Step 1). Subsequently, the front-end devices synthesize challenges according to the received parameters and deliver them to the user (Step 2). After receiving the challenges, the user makes expressions, such as smiling blinking, as responses. The sensors in the front-end devices capture such responses and encode them (Step 3). Either in real time or in post processing, the front-end devices send the captured responses to the liveness detection module in the back-end server (Step 4). The liveness detection module gathers all decoded data and checks whether the user is an actual human being. If so, the liveness detection module selects some faces among all the responses and delivers them to the face recognition module to determine the identity of the user (Step 5).

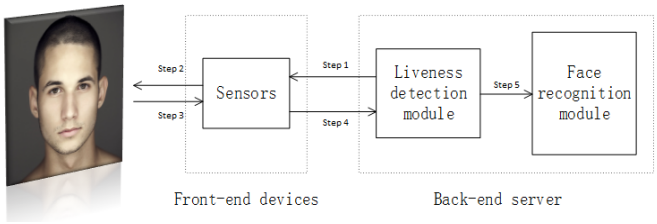


Fig. 1: A typical face authentication system.

B. Attacks and Solutions on Liveness Detection

In recent years, plenty of attacks have been developed to exploit the flaw that face recognition algorithms cannot determine whether a photograph taken by the front-end camera is captured from a real face, even if the recognition accuracy of some has exceeded human beings. In this study, we divide attacks into four categories and organize them as a tree, known as the attack tree, which is displayed in Fig 2. We first separate attacks into two categories: *static* and *dynamic*. *Static* attacks refer to the use of static objects, such as photographs, plastic masks, and paper, as well as transformations of these objects (e.g., folding, creating holes through them, and assembling them into certain shapes). Attacks using dynamic or partially dynamic objects are categorized into dynamic branch. Subsequently, we separate attacks into four subcategories: two-dimensional (2D) static, three-dimensional (3D) static, 2D dynamic, and 3D dynamic. The 3D branches refer to attacks that use stereo objects, including sculptures, silicone masks,

and robots. More precisely, these objects must have notable stereo characters of human faces, such as a prominent nose, concave eye sockets and salient cheekbones; otherwise, the attacks are categorized into *2D* branches. Organized by this attacking tree, a brief review of relative attacks and solutions is presented below.

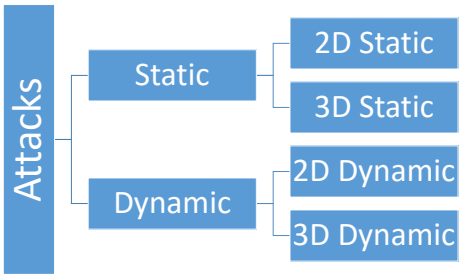


Fig. 2: The attacking tree.

In the *2D static* branch, photograph-based attacks are the predominant form of attack. They are easily launched and effective at compromising primary recognition algorithms. Tracking eye movement was the first method proposed to counter such attacks. Jee et al. [9] proposed a method for detecting eyes in a sequence of facial images. Variations around the eyes are calculated, and whether the face is real is determined. Their basic assumption is that blinking and the uncontrolled movements of pupils are human nature behaviors. However, this method cloud be compromised by an adversary wearing a mask with eyeholes. A similar idea exploiting Conditional Random Fields (CRFs) was proposed by Sun et al. [23]. Its limitation is the same. Subsequently, lip movement and lip reading methods were developed by Kollreider et al. [14] for liveness detection. However, their method can also be fooled using carefully transformed photographs.

To distinguish faces from photographs, Li et al. [16] leveraged the fact that faces and photographs have different appearances in frequency space. They conducted 2D Fourier spectral analysis to extract the high-frequency components of input images; faces contain more components in the high-frequency region than do photographs. However, adversaries can print a high-resolution photograph to bypass this method. Jukka et al. [18] observed that printed photographs are pixelized. That is, a face has more details than a photograph. Thus, they used a support vector machine (SVM) to extract microtextures from the input image. Later, Kim et al. [11] leveraged a more powerful texture descriptor, Local Binary Pattern (LBP) to enhance performance. They additionally analyzed the information residing in both the low- and high-frequency regions. However, all these types of solutions have a common drawback: low robustness. Motion blur or noise from the environment impairs their performance. Moreover, these methods are useless against screen-based attacks [5].

Because of strategies designed to protect against 2D attacks, adversaries have attempted to exploit 3D objects in 3D static attacks. Thus, researchers have developed novel methods to defend against these attacks also. Lagorio, et al., [15] proposed a method to detect 3D facial features. They employed a two- camera device to ascertain the surface curvature of the subject seeking authentication. If the surface curvature is low, the subject is judged to be malicious. Although the accuracy is almost 100%, the sophisticated device required is expensive and the computational cost is unacceptable. Furthermore, Wang, et al., [24] leveraged a one-camera, face alignment algorithm to ascertain 3D shape on the basis that forged faces are usually flatter than real faces. However, this method performed unsatisfactorily when applied to spectacle-wearers because of the limited capability of the face alignment algorithm.

In response to technological developments, adversaries must in turn develop more sophisticated attacks. One method is pasting stereo materials onto photographs. In contrast, researchers have developed practical and efficient methods to counter these threats, with the help from developments in computer vision. The fundamental idea behind these methods is that adversaries cannot manipulate static objects to simulate instructed expressions, even if these objects are similar to the human face. Thus, a common challenge-response protocol has been adopted, whereby users are asked to make expressions as the instructions, including happiness, despair, and surprise. Such systems subsequently compare the captured video with stored data.

However, more powerful *2D dynamic* attacks have been developed, in which adversaries have exploited advanced deep learning models and personal computers with powerful processors. These attacks work by merging a victim's facial characteristics with photographs of the victim and using these forged photographs to bypass the face recognition algorithm. Furthermore, even if this operation requires time, adversaries can prepare photographs beforehand and launch offline attacks, sending forged photographs to an online authentication system.

To counter these new 2D dynamic threats, some solutions have been proposed. Bao et al.. [1] introduced a method based on optical flow technology. The authors found that the motions of 2D planes and 3D objects in an optical flow field are the same in translation, rotation, and moving, but not in swing. They used this difference to identify fake faces. Unfortunately, two drawbacks undermine this method: first, an uneven object will fail it; second, the method does not consider variation in illumination. Kollreider, et al., [13] developed a method for detecting the positions and velocities of facial components by using model-based Gabor feature classification and optical flow pattern matching. However, its performance was impaired when keen edges were present on the face (e.g., spectacles or a beard). The authors admitted that the system is only error-free if the data contain only horizontal movements. Findling et al. [8] achieved liveness detection by combining camera images and movement sensor data. They captured multiple views of a face while a smartphone was moved. Li et al. [17] measured the consistency in device movement detected using inertial sensors and changes in the perspective of a head captured on video. However, both methods were demonstrated to be compromised by Xu et al. [27], who constructed virtual models of victims on the basis of publicly available photographs of them.

Less adversaries have attempted to launch *3D dynamic* attacks. They can reconstruct a 3D model of a victim's

face [27] in virtual settings but hardly fabricate them in real scenes. We illustrate the difficulties in launching 3D dynamic attacks using the following three examples: First, building a flexible screen that can be molded into the shape of a face is expensive and may fail because the reflectance from a screen differs from that of a face. Second, 3D printing a soft mask is impractical, being limited by the printing materials available (see Section V-D for a fuller explanation). Third, building an android is infeasible and intricate and would involve face animation, precision control, and skin fabrication. Additionally, building an android is costly, particularly a delicate android face.

On the basis of the above discussion, we observe that the current threats are principally 2D dynamic attacks because static attacks have been effectively neutralized and 3D dynamic attacks are hard to launch.

III. Adversary Model

In this section, we present our proposed adversary model and assumptions.

We assume adversaries' goal is to bypass the face authentication systems by impersonating victims, and the objective of our proposed methods is to raise the bar for such successful attacks significantly. As will be demonstrated in the limitation part (section IX), powerful adversaries could bypass our security system, but the cost would be much higher than is currently the case. Particularly, they need to purchase or build special devices that can do all of the following operations within the period when the camera scanning a single row: (1) capture and recognize the randomized challenges, (2) forge responses depending on the random challenges, and (3) present the forged responses. For this, adversaries require high-speed cameras, powerful computers, high-speed I/Os, and a specialized screen with fast refreshing rate, etc. Therefore, it is difficult to attack our system.

Adversaries can also launch *3D dynamic* attacks, such as undergoing cosmetic surgery, disguising their faces, or coercing a victims twin into assisting them. However, launching a successful *3D dynamic* attack is much more difficult than using existing methods of MFF attack; crucially, identifying such an attack would be challenging even for humans and would constitute a Turing Test problem, which is beyond the scope of this paper. But in either case, our original goal is achieved by having increased the bar for successful attacks significantly.

Our method relies on the integrity of front-end devices; that is, that the camera and the hosting system that presents random challenges and captures responses have not been compromised. If this cannot be guaranteed, adversaries could learn the algorithm used to generate random challenges and generate fake but correct responses beforehand, thus undermining our system. We believe that assuming the integrity of front-end devices is reasonable in real-world settings, considering that in many places the front-end devices can be effectively protected and their integrity guaranteed (e.g., ATMs and door access controls). We cannot assume or rely on the integrity of smartphones, however. Our proposed techniques are general and can easily be deployed on different hardware platforms, including but not limited to smartphones. For simplicity, we choose to build a prototype and conduct evaluations on smartphones, but this is only for demonstration purposes. If the integrity of a smartphone can be guaranteed, by using a trusted platform module or Samsung KNOX hardware assistance, for example, our techniques can be deployed on them; otherwise this should be avoided and the proposed techniques are not tied to smartphones.

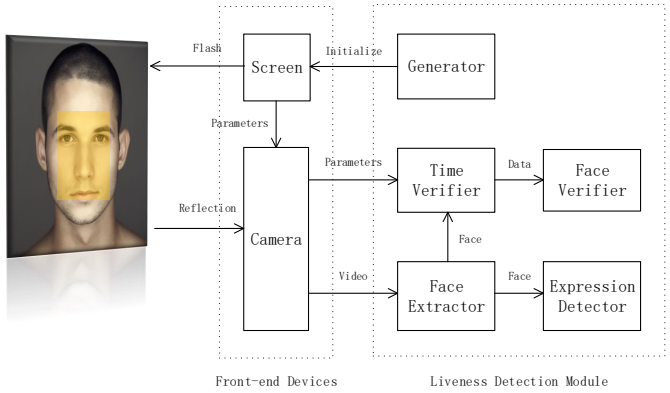


Fig. 3: Architecture of Face Flashing.

IV. Face Flashing

Face Flashing is a challenge-response protocol designed for liveness detection. In this section, we elaborate its detailed processes and key techniques to leverage flashing and reflection.

A. Protocol Processes

The proposed protocol contains seven components, which are illustrated in Fig 3, and eight steps are required to complete once challenge-response procedure where the challenge is flashing light and the response is the light reflected from the face.

- **Step 1: Generation of parameters.** Parameters are produced by the *Generator* of the *Liveness Detection Module* running on the back-end server, which works closely with the front-end devices. Such parameters include seed and N. Seed controls how random challenges are generated, and N determines the total number of challenges to be used. Communication of parameters between the back-end server and front-end devices is protected by standard protocols such as HTTP and TLS.
- **Step 2: Initialization of front-end devices.** After receiving the required parameters, the front-end devices initialize their internal data structures and start to capture videos from the subject being authenticated by turning on the *Camera*.
- **Step 3: Presentation of a challenge.** Once initialized, the front-end devices begin to generate challenges according to the received parameters. Essentially, a challenge is a picture displayed on a screen during one refresh period; light is emitted by the screen onto the subjects face. The challenge can be of two types:

a background challenge, which displays a pure color, and a lighting challenge, which displays a lit area over the background color. More details are specified in subsequent sections.

- **Step 4: Collection of response.** The response is the light that is reflected immediately by the subjects face. We collect the response through the camera that has already been activated (in Step 2).
- **Step 5: Repetition of challenge-response.** Our protocol repeats Step 3 and 4 for N times. This repetition is designed to collect enough responses to ensure high robustness and security so that legitimate users always pass whereas adversaries are blocked even if they accidentally guess out the correct challenges beforehand.
- **Step 6: Timing verification.** Timing is the most crucial security guarantee provided by our protocol and is the fundamental distinction between genuine and fake responses. Genuine responses are the light reflected from a human face and are generated through a physical process that occurs simultaneously over all points and at the speed of light (i.e., zero delay). Counterfeit responses, however, would be calculated and presented sequentially, pixel by pixel (or row by row), through one or more pipelines. Thus, counterfeit responses would result in detectable delays. We detect delays among all the responses to verify their integrity.
- **Step 7: Face verification.** The legality of the face is verified by leveraging neural network that incorporates with both the shape and textual characters extracted from the face. This verification is necessary because without it our protocol is insufficiently strong to prevent from MFF attacks, and face verification prolongs the time required by adversaries to forge a response, which makes the difference from benign response more obvious. The details are provided in Section IV-B.
- **Step 8: Expression verification.** The ability to make expressions indicates liveness. We verify this ability by ascertaining whether the detected expression is the one requested. Specifically, technology from [22] is embedded in our prototype for detecting and recognizing human expressions.

Details of Step 8 are omitted in this paper so that we can focus on our two crucial steps: timing and face verifications. However, expression detection has been satisfactorily developed and is critical to our focus. Additionally, Step 8 is indispensable because it integrates our security boundary, which is elucidated in Section V. The face extraction detailed in the next section is designed so that our two verification techniques are compatible with this expression detection.

B. Key Techniques

The security guarantees of our proposed protocol are built on the timing as well as the unique features extracted from the reflected lights. In the followings, we will first introduce the model of light reflection, then our algorithm for extracting faces from video frames, and verifications on time and face.

*1) **Model of Light Reflection**:* Consider an image $I_{rgb} = \{I_r, I_g, I_b\}$ that is taken from a linear RGB color camera with black level corrected and saturated pixels removed. The value of $I_c, c \in \{r, g, b\}$ for a Lambertian surface at pixel position x is equal to the integral of the product of the illuminant spectral power distribution $E(x, \lambda)$, the reflectance $R(x, \lambda)$ and the sensor response function $S_c(\lambda)$:

$$I_c(x) = \int_\Omega E(x, \lambda)R(x, \lambda)S_c(\lambda)d\lambda, c \in \{r, g, b\}$$

where λ is the wavelength, and Ω is the wavelength range of all visible spectrum supported by camera sensor. From the Von Kries coefficient law [3], a simplified diagonal model is given by:

$$I_c = E_c \times R_c, c \in \{r, g, b\}$$

Exploiting this model, by controlling the outside illuminant E, we can get the reflectance of the object. Specifically, when E_c for x and y are the same, then

$$\frac{I_c(x)}{I_c(y)} = \frac{R_c(x)}{R_c(y)}, c \in \{r, g, b\} \quad (1)$$

This means the lights captured by camera sensor at two different pixels x and y are proportional to the reflectance of that two pixels.

Similarly, for the same pixel point x, if applying two different illuminant lights E_{c1} and E_{c2}, then:

$$\frac{I_{c1}(x)}{I_{c2}(x)} = \frac{E_{c1}(x)}{E_{c2}(x)}, c1, c2 \in \{r, g, b\} \quad (2)$$

In other words, the reflected light captured by the camera in a certain pixel is proportional to the incoming light of the same pixel.

Implications of above equations. Eq.(1) and Eq.(2) are simple but powerful. They are the foundations of our liveness detection protocols. Eq.(1) allows us to derive relative reflectance for two different pixels from the proportion of captured light from these two pixels. The reflectance is determined by the characteristics of the human face, including its texture and 3D shape. Leveraging Eq.(1), we can extract these characteristics from the captured pixels and further feed them to a neural network to determine how similar the subject's face is to a real human face.

Eq.(2) states that for a given position, when the incoming light changes, the reflected light captured by the camera changes proportionally, and crucially, such changes can be regarded as "simultaneously" to the emission of the incoming light because light reflection occurs at the speed of light. Leveraging Eq.(2), we can infer the challenge from the current received response and detect whether a delay occurs between the response and the challenge.

*2) **Face Extraction**:* To do our verifications, we need to locate the face and extract it. Furthermore, our verifications must be performed on regularized faces where pixels in different frames with the same coordinate represent the same point on the face. Concretely, when a user's face is performing expressions as instructed, it produces head movements and hand tremors. Thus, using only face detection technology is insufficient; we must also employ a face alignment algorithm

that ascertains the location of every landmark on the face and neutralizes the impacts from movements. Using the alignment results, we can regularize the frames as we desired, and the regularized frames also ensure that our verifications are compatible with the expression detector.

First, We designed Algorithm 1 to quickly extract the face rectangle from every frame. In Algorithm 1, $track(\cdot)$ is our face tracking algorithm [7]. It uses the current frame as the input and employs previously stored frames and face rectangles to estimate the location of the face rectangle in the current frame. The algorithm outputs the estimated rectangle and a confidence degree, ρ. When it is small ($\rho < 0.6$), we regard the estimated rectangle as unreliable and subsequently use $detect(\cdot)$, our face detection algorithm [28], to redetect the face and ascertain its location. We employ this iterative process because the face detection algorithm is precise but slow, whereas the face tracking algorithm is fast but may lose track of the face. Additionally, the face tracking algorithm is used to obtain the transformation relationship between faces in adjacent frames, which facilitates our evaluation of robustness (Sec VI-D).

Algorithm 1 Algorithm to extract the face.

INPUT: $Video$
OUTPUT: $\{F_j\}$
for $frame$ in $Video$ **do**
  $Rect, \rho$ = track($frame$)
  if $Rect = \emptyset$ or $\rho < 0.6$ **then**
    $Rect$ = detect($frame$)
    $Rect \rightarrow$ track($\cdot$)
  end if
  F_j = $frame(Rect)$
end for

After obtaining face rectangles, $\{F_j\}$, we exploit face alignment algorithm to estimate the location of 106 facial landmarks [29] on every rectangle. The locations of these landmarks are shown in Figure 4.

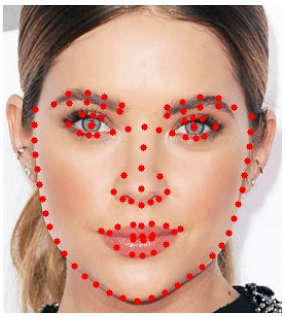


Fig. 4: 106 landmarks.

Further, we use alignment results to regularize every rectangle. Particularly, we formalize the landmarks on j-th face as $L_j = (l_1, l_2, \cdots, l_{106})$, where l_i denotes $(x_i, y_i)^\top$, the coordinates of i-th landmark. And, we calculate the transformation matrix T_j by:

$$\begin{aligned} T_j &= \text{argmin}_T ||T\tilde{L}_j - L_{mean}||^2 \\ \text{where} \quad L_{mean} &= \frac{\sum_j L_j}{\sum_j 1} \\ \tilde{L}_j &= \begin{bmatrix} x_1 & x_2 & \cdots & x_{106} \\ y_1 & y_2 & \cdots & y_{106} \\ 1 & 1 & \cdots & 1 \end{bmatrix} \end{aligned}$$

where T is a 3×3 matrix contains rotation and shifting coefficients. We select the best T as T_j that minimizes the L_2 distance between the regularization target T_{mean} and $\tilde{L}_j$, the homogeneous matrix of the coordinate matrix. After that, we regularize the j-th frame by applying the transformation matrix T_j to every pair of coordinates and extract the centering 1280x720 rectangle containing the face. For the sake of simplicity, we use "frame" to represent these regularized frames containing only the face [1].

*3) **Timing Verification**:* Our timing verification is built on the nature of how camera and screen work. Basically, both of them follow the same scheme: refreshing pixel by pixel. Detailedly, after finishing refreshing one line or column, they move to the beginning of next line or column and perform the scanning repeatedly. We can simply suppose an image is displayed on screen line by line and captured by camera column by column, ignoring the time gap between refreshing adjacent pixels within one line or column that is much smaller than the time needed to jump to the next line or column. In other words, as to update any specific line on the screen, it has to wait for a complete frame cycle until all other lines have been scanned. Similarly, when a camera is capturing an image, it also has to wait for a frame cycle to refresh a certain column.

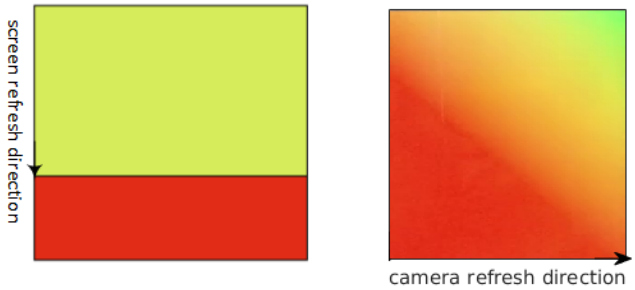
One example is given in Fig 5 to better explain the interesting phenomenon that is leveraged for our timing verification. Fig 5a shows a screen that is just changing the displaying color from Red to Green. Since it is scanning horizontally from top to bottom, the upper part is now updated to Green but the lower part is still previous color Red. The captured image of a camera with column scanning pattern from left to right is shown in Fig 5b, which shows an obvious color gradient from Red to Green [2].

To transform this unique feature into a strong security guarantee, the appropriate challenges must be constructed and verified to ensure the consistency of responses. In practice, we construct two types of challenge to be presented on front-end screen: one is the background challenge displaying a single color, and the other is the lighting challenge displaying a belt of different color on the background color. The belt of the different color from background is called the *lighting area*, and one example is shown in Fig 6a, where the background color is Red while the lighting area is Green.

To verify the consistency in responses, we defined another concept called *Region of Interest* (ROI), which is the region that the camera is scanning when the front-end screen is

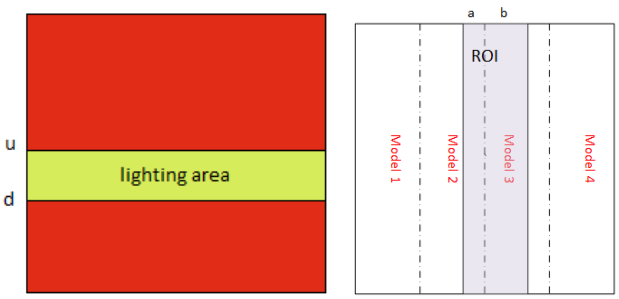
[1] Since we just implemented those existing algorithms on face tracking, detection and alignment, we will not provide further details about them and interested readers can refer to original papers.

[2] Similar but a little bit different color patterns can also be observed on cameras with row scanning mode. Column scanning mode is used here it is easier to understand.



(a) screen refreshing (b) camera refreshing

Fig. 5: Working schemes of screen and camera.



(a) lighting challenge (b) captured frame

Fig. 6: Example of lighting area and calculation of ROI.

displaying the lighting area. The location of ROI is calculated as followings:

- Calculate t_u, the start time to show the lighting area.

$$t_u = t_{begin} + \frac{u}{rows} * t_{frame} \tag{3}$$

where u is the upper bound of lighting area, $rows$ is the number of rows contained in one frame, t_{begin} is the start time to show the current frame, and t_{frame} is the during time of one frame.

- Find the captured frame which recording period covers t_u. Say the k-th captured frame.
- Calculate the shift, l, against the first column of k-th captured frame.

$$l = cols * \frac{t_u - ct_k}{ct_{frame}} \tag{4}$$

where $cols$ is the number of columns contained in one captured frame, ct_k is the start time to exposure the first column of k-th capture frame, and ct_{frame} is the exposure time of one captured frame.

After finding the location of ROI, we distill it by applying Eq.(2) on every pixel between the response of lighting challenge and background challenge. Two applied results are demonstrated on Fig 7. Now, the consistence can be verified. We check whether the lighting area can be correctly inferred from the distilled ROI. If it cannot, the delay exists and this response is counterfeit.

To infer the lighting area, we build 4 linear regression models handling different part of captured frame (Fig 6b). Each model is fed a vector, the average vector reduced from corresponding part of ROI, and estimates the location of $\frac{u+d}{2}$ independently. Next we gather estimated results according to the size of each part. An example is shown on Fig 6b where the ROI is separated into 2 parts: the left part contains a columns and the right part contains b columns. The gathered result, $\hat{y}$, is calculated as following.

$$\hat{y} = \frac{a \times m_2 + b \times m_3}{a + b} \tag{5}$$

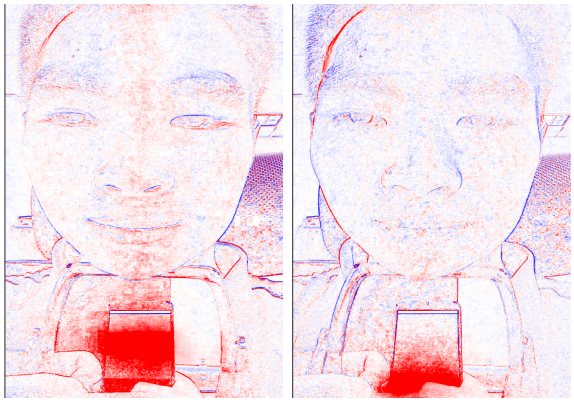
where m_2 and m_3 denote the estimated result made by model 2 and model 3 respectively.

The final criteria of consistence is accumulated from $\hat{y}_i$, the gathered result of i-th captured frame, as following:

$$\begin{aligned} d_i &= \hat{y}_i - \tfrac{u_i+d_i}{2} \\ mean_d &= \tfrac{\sum_{i=1}^{n} d_i}{n} \\ std_d^2 &= \tfrac{\sum_{i=1}^{n}(d_i - mean_d)^2}{n-1} \end{aligned} \tag{6}$$

We finally check whether $mean_d \times std_d$ is smaller than $exp(Th)$, where Th is a predefined threshold.

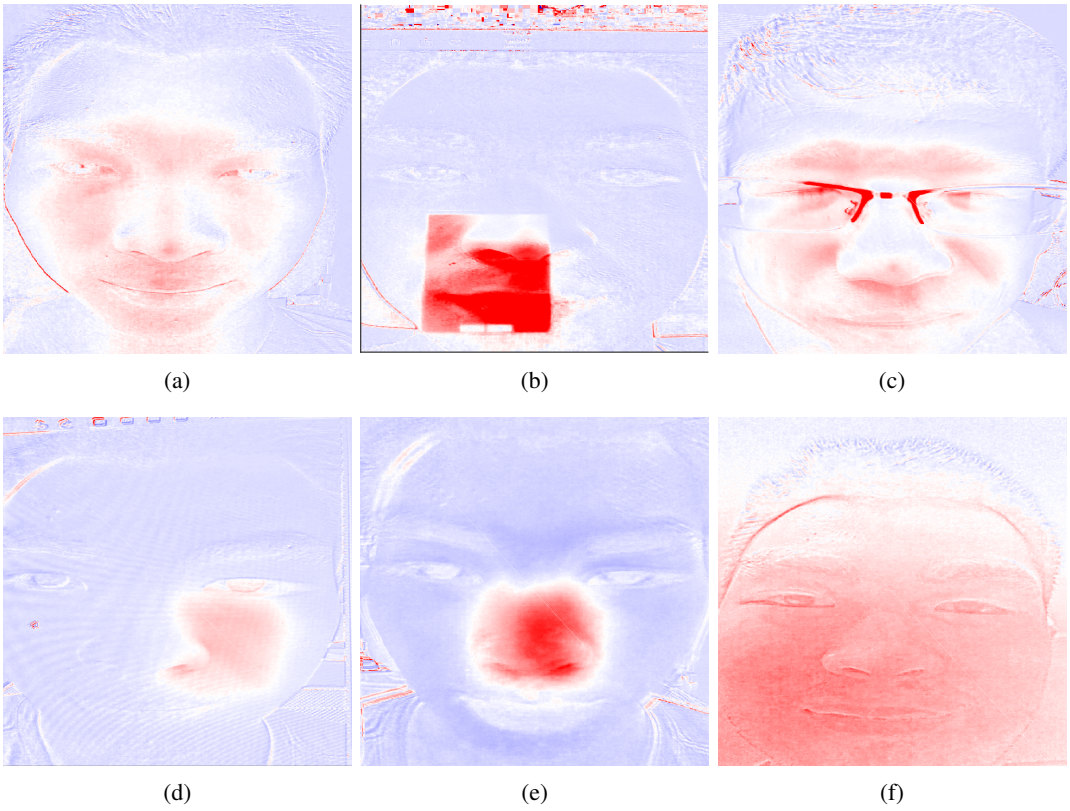
Note that legitimate responses are consistent with our challenges and will produce both small $mean_d$ and std_d. Adversarial responses will be detected by checking our final criteria. An additional demo was illustrated on Fig 7 to explain visually how the lighting area affects the captured frame.



(a) light middle area (b) light bottom area

Fig. 7: Effect of lighting area. In the bottom of both pictures, these are mirrors showing the location of corresponding lighting area.

*4) **Face Verification**:* After preprocessing, we get a sequence frames with vibration removed, size unified and color synchronized. Further, we use Eq.(1) to generate the midterm result from the responses of a background challenge: First, we randomly choose a pixel on the face as the anchor point; then, we divide all the pixels by the value of that anchor point. Some midterm results are shown on Fig 8.



(a) (b) (c)

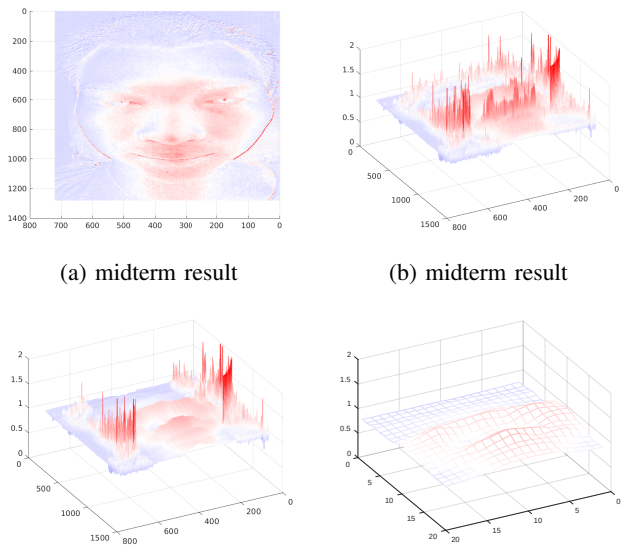
(d) (e) (f)

Fig. 8: Examples of midterm results. (a) and (c) are captured from real human faces, (b) is captured from an iPad's screen, (d) and (e) are captured from a LCD monitor, and (f) is captured from a paper.

Without any difficulty, we can quickly differentiate results of real human faces from fake ones. This is because real human faces have uneven geometry and textures, while other materials, like monitor, paper or iPad's screen, do not have. Based on this observation we developed our face verification techniques, as described below.

- Step 1: abstract. We vertically divide the face into 20 regions. In every region, we further reduce the image to a vector by taking the average value. Next, we smooth every vector by performing polynomial fitting of 20 degrees with minimal 2-norm deviation. After that, we will derive images like Fig 9c.
- Step 2: resize. We pick out facial region and resize it to a 20x20 image by bicubic interpolation. An example is shown on Fig 9d.
- Step 3: verify. We feed the resized image to a well-trained neural network, and the decision will be made then.

The neural network we used contains 3 convolution layers with a pyramid structure, which effectiveness was sufficiently proved in Cifar-10, a dataset used to train the neural network to classify 10 different objects. In Table I, we show the architecture of our network and the parameters of every layer.



(a) midterm result (b) midterm result

(c) abstract result (d) resize result

Fig. 9: Face verification.

V. Security Analysis

In this section, we present the security analysis of Face Flashing. First, we abstract the mechanisms behind Face Flashing as a challenge-response protocol. Second, we analyze the security of two main parts in our protocol: timing verifications and face verification. Finally, we demonstrate how Face Flashing defeats three typical advanced attacks.

TABLE I: Architecture of Neural Network.

input size	layer type	stride	padding size
20x20x3	conv 5x5	1	0
16x16x16	conv 3x3	1	1
16x16x16	pool 2x2	1	0
8x8x16	conv 3x3	1	1
8x8x32	pool 2x2	1	0
1x512	inner product	0	0

It is certain that Face Flashing can defeat static attacks, as the expression detector, one component of our system, is sufficient to defeat them. Specifically, static materials cannot make expressions according to our instructions in time (e.g., 1 second) and attacks using them will be failed by expression detector. Besides, we conduct a series of experiments in Section VI to demonstrate that the expression detector can be correctly integrated with our other verifications. Therefore, the main task of our security analysis is to show that Face Flashing can defeat dynamic attacks.

A. A Challenge-Response Protocol

Face Flashing is a challenge-response protocol whose security guarantees are built upon three elements: unpredictable random challenges, hardly forged responses, and the effective response verifications.

The Challenges. Our challenge is a sequence of carefully-crafted images that are generated at random. Since the front-end devices are assumed to be well protected, adversaries could not learn the random values. Besides, a verification session consists of tens of challenges. Even if the adversary can respond a right challenge by chance, it is unlikely for him to respond to a sequence of challenges correctly.

The Responses. There are two important requirements for the responses: First, the response must be easily generated by the legitimate users, otherwise it may lead to usability problems or even undermine the security guarantee (e.g., if adversaries can generate fake responses faster than legitimate users). Secondly, the responses should include essential characteristics of the user, which are hard to be forged.

Face Flashing satisfies both requirements. The response is the reflected light from the human face, and the user needs to do nothing besides placing her face against the camera. More importantly, such responses, in principle, are generated at the speed of light, which is faster than any computational process. Besides, the response carries unique characteristics of the subject, such as the reflectance features of her face and uneven geometry shapes, which are physical characteristics of human faces that are inherently different from other media, e.g., screens (major sources of security threats).

Response Verification. We use an in-depth defense strategy to verify the responses and detect possible attacks.

- First, timing verification is used to prevent forged responses (including replay attacks).
- Second, face verification is used to check if the subject under authentication has a specific shape similar to a real human face.
- Third, this face-like object must be regarded as the same person with the victim by the face recognition module (orthogonal to liveness detection).

Considering the pre-excluded static object, it is very hard for adversaries to fabricate such a thing satisfying 3 rules above simultaneously. In general, Face Flashing builds a high bar in front of adversaries who want to impersonate the victim.

B. Security of Timing Verification

The goal of the timing verification is to detect the delay in the response time caused by adversaries. Before further analysis, we emphasize two points should be considered.

- First, according to the design of modern screens, the adversary cannot update the picture that is being displayed on the screen at the arbitrary time. In other words, the adversary cannot interrupt the screen and let it show the latest generated response before the start of next refreshing period.
- Second, the camera is an integral device which accumulates the light during his exposure period. And, at any time, within an initialized camera, there always exists some optical sensors are collecting the light.

For sake of clarity, we assume the front-end devices contain a 60-fps camera and 60-Hz screen. On the other side, the adversary has a more powerful camera with 240-fps and screen with 240-Hz. Under these settings, we construct a typical scenario to analyze our security, which time lines are shown on Fig 10.

In this scenario, the screen of the front-end device is displaying the i-th challenge, and the adversary aims to forge the response to this challenge. The adversary may instantly learn the location of lighting area of the challenge after t_u. While she cannot present the forged response on her screen until v_k, due to the nature of how the screen works. Hence, there is a gap between t_u and v_k. Recalling our method described in Section IV-B3, during the gap, some columns in the ROI have already completed the refreshing process. In other words, these columns' image will not be affected by the forged response displaying on the adversary's screen during v_k to v_{k+1}. We name this phenomenon as *delay*.

When *delay* happens, our camera will get an undesired response inducing four linear regression models to do deviated estimation about the location of lighting area. Besides, the standard deviation of these estimations will increase, for two reasons:

- The adversary's screen can hardly be synchronized with our screen. Particularly, it is different even the length of adjacent refreshing periods. Hence, the $delay$ is unstable, so as the estimations.
- The precision of forging will be affected by the internal error of adversaries' measurement about time. This imprecision will be amplified again by our camera, which fluctuates the estimations.

In other words, if the adversary reduces $mean_d$ by displaying the carefully-forged response, she will simultaneously

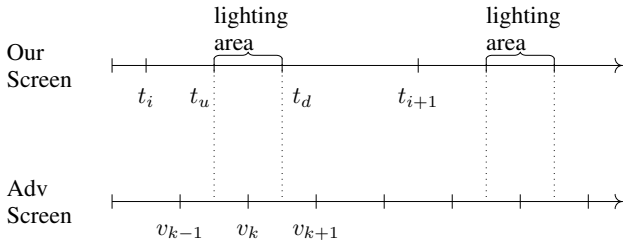


Fig. 10: Security analysis on time.

increase std_d. On the other hand, if the adversary does nothing to reduce std_d, she will significantly enlarge $mean_d$. While for a benign user, the *delay* will not happen, the discordance between our camera and screen can be solved by checking the timestamps afterward, and both the accumulated $mean_d$ and std_d will be small, according to our verification algorithm.

In summary, we detect the delay by estimating the deviation. And the effectiveness of our algorithm provides a strong security guarantee on the timing verification.

C. Security of Face Verification

Our face verification abstracts the intrinsic information of shape through a series of purification. And we feed this information to a well-designed neural network.

If the adversary aims to bypass the face verification, there are two conundrums that need to be resolved. First, the adversary needs to conceal the specular reflection of the plain screen. Particularly, during the authentication procedure, we require the user to hold the phone so their face can occupy the entire screen. The distance, as we measured, is about 20-cm. In this short distance, the specular reflection is severe. In Fig 8b, we demonstrate the result captured from a screen without any covering sheet. Even covered by a scrub film (Fig 8d), the screen's specular reflection is still intense.

Second, the forged object must have similar geometry shape with human faces. More precisely, its abstract result should like a transpose of "H" (Fig 9c). And this stereo object needs to make expression according to our instructions. Even if the adversary can achieve these, there is no promise they can deceive our strong neural network modal every time. And there is no chance for the adversary to generate a response with low quality. The high recall of our model will be demonstrated in next section.

The above two conundrums provide the security guarantee on face verification.

D. Security against typical attacks

Obviously, Face Flashing can defeat traditional attacks like photo-based attacks. Here we discuss its defenses against three typical advanced attacks:

Offline Attacks. An offline attack is to record responses of previous authentications, and replay them to attack the current authentication. However, this attack is impossible to fool our protocol. First, the hitting possibility is small, as we require responses match all the challenges. Concretely, if we use 8 different colors and present 10 lighting challenges, the hitting possibility will be less than 10^{-9}. Second, even if adversaries have successfully guessed the correct challenge sequence, displaying responses legitimately is difficult. Because displaying on screens will produce the intensely specular reflection that is easily detectable, and displayed by projecting responses onto a forged object leads to high std_d that also can be detected, as adversaries cannot precisely predict the length of every refreshing period of the screen.

MFF Attacks. An MFF attack is to forge the response by merging victim's facial information and the currently received challenge. However, this attack is also useless, because it is hard to deceive our timing and face verifications simultaneously. First, to deceive our face verification needs forging high-quality responses which is difficult and time-consuming. Particularly, high-quality forgery requires reconstructing the 3D model of victim's face and simulating the reflection process. Second, to deceive our timing verification needs to complete the above forgery quickly. Actually, the available time is $\frac{1}{240}/2$ second for attacking a 60 Hz screen (Section VI-B). Third, even if adversaries can quickly produce a perfectly forged response, displaying the response is not allowed (see the preceding paragraph).

3D-Mask Attacks. A 3D-mask attack is to wear a 3D mask to impersonate the victim. However, this attack is impractical. First, this attack needs to build an accurate mask that can fool our face recognition module, which is difficult[3]. Second, the legitimate mask is hard to be 3D printed. As the printed mask needs to have the similar reflectance of human skin and be so flexible that adversaries can wear it to make instructed expressions. While, the available 3D printed materials are non-flexible under the requirement of Fused Deposition Modeling (FDM), the prevalent 3D print technology. Besides, the smallest diameter of available nozzles is 0.35mm that will produce coarse surfaces, and coarse surfaces can be distinguished from human skin.

In sum, Face Flashing is powerful to defeat advanced attacks, especially attacks similar to the ones mentioned above.

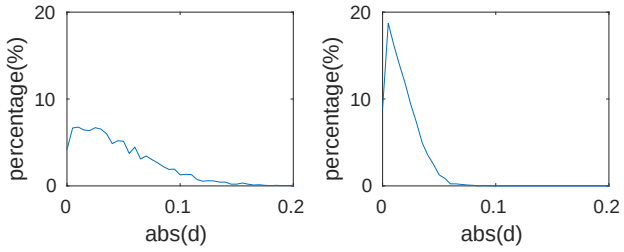
VI. Implementation and Evaluation

In this section, we introduce the source of our collected data at the beginning, then present implementations and evaluations of timing and face verifications, followed by the evaluation on robustness. Finally, we give the computational and storage cost when deploying our system on a smartphone and the back-end server.

A. Data Collection

We have invited 174 participants including Asian, European and African. Among all participants, there are 111 males and 63 females with ages ranging from 17 to 52. During the experiment, participants were asked to hold a mobile phone facing to their face and make expressions such as smiling, blinking or moving head slightly. A button was located at the bottom of the screen so that participants can click it to start

[3]Even though there is an existing study implying it is possible [19], performing it in real is not easy.



(a) mean=0.046, std=0.035 (b) mean=0.012, std=0.013

(c) mean=0.020, std=0.015 (d) mean=0.060, std=0.045

Fig. 11: Performance of 4 regression models. (a)-(d) shows performance of model 1-4 respectively.

(and stop) the authentication/liveness detection process. When started, the phone performs our challenge-response protocol and records a video with its front camera. And, once started, that button will be disabled for three seconds to ensure that every captured video contains at least 90 frames.

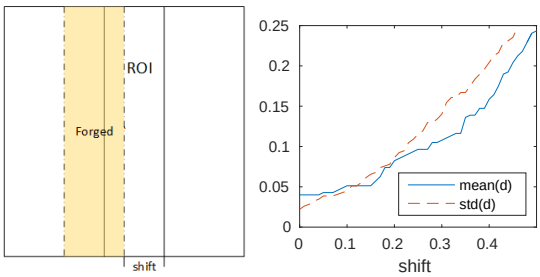
In total, we collect 2274 raw videos under six different settings (elaborated in Section VI-D). In each scenario, we randomly select 50 videos to form the testing data set, and all other videos then belong to the training data set.

B. Timing Verification

In our implementation of timing verification, we set the height of lighting area in every lighting challenge to a constant, i.e., $u - d = 1/4$, where the height of the whole screen is 1. And we use an open source library, *LIBLINEAR* [6], to do the regression with $L2$-loss and $L2$-regularization, where the Th is set to -5.

We trained four regression models on the training set mentioned above, and their performances over the testing data set are shown on Fig 11. It shows that performances of model 1 and 4 are relatively poor which is reasonable in fact, because both models handle two challenging areas (refer to Fig 6) where the responses are weak and the keen edges also impair the results.

To evaluate its capability on defending against attacks, we feed forged areas (see Fig 12a) to these regression models, and observe the results. It turns out that when enlarging the $shift$ between real ROI and forged area, the estimation deviation increases. In Fig 12b, we illustrated the relationship between estimated $mean_d$ and std_d under different values of $shift$, while regularizing the width of ROI as 1. The figure shows that when $shift$ is less than 0.1, the estimation error of $mean_d$ and std_d is very small. But when the shift is 0.5, the estimation error is around $1/4$. In other words, when increasing $shift$



(a) (b)

Fig. 12: Attack simulation

to the half of ROI's width, the estimated deviation could be larger than the height of the lighting area, which states that adversary's opportunity window (i.e., $shift$) for a successful attack is pretty small, and our method can reliably detect such attacks. Concretely, the acceptable delay for a benign response is less than $\frac{1}{240}/2$ second for a 60 Hz screen.

Further, we investigated the delays under a real-world setting (shown in Fig 13). In this experiment, we used two devices: A is the authenticator (a Nexus 6 smartphone in this example), and B is the attacker (a laptop that will reproduce the color displayed on smartphone by simply showing the video captured by its front camera). When the experiment begins, the smartphone starts to flash with random colors, and record whatever is displayed on laptop screen at the same time, then calculate the delay needed by attackers to reproduce the same color. The same procedure will be repeated to calculate the delays by replacing the laptop with a mirror.

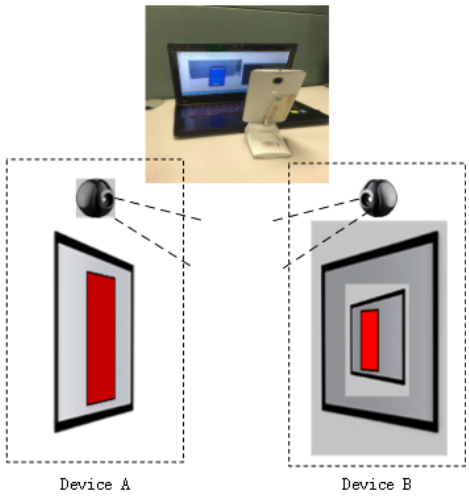
Fig 13b shows the results where the blue bars are mirror's delays while the red bars are the laptop's delays. The difference between the delays means that if adversaries had used devices other than mirrors to reproduce the reflected colors (i.e. responses), there should be significant delays. This is actually one of our major technical contribution to use light reflections instead of human expressions and/or actions as the responses to given challenges, and it can give a clear and strong timing guarantee to differentiate genuie and fake responses.

C. Face Verification

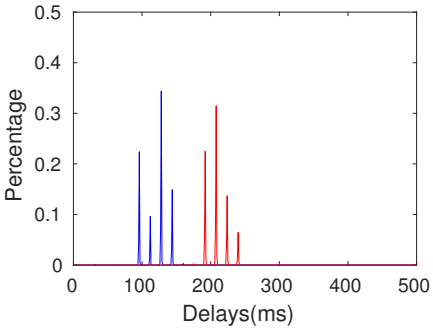
We use *Caffe* [10], an open source deep learning framework, to train our neural network model used for face verification. The preliminary parameters are listed below: learning policy is set to "multistep", base learning rate is 0.1, gamma is 0.1, momentum is 0.9, weight decay is 0.0001 and the max iteration is 64000.

We first build a set of adversarial videos in order to train the model. These videos are made by recording the screen that is replaying the raw video. There are 4 different screens are recorded (Table II).

We take those frames in malicious videos as our negative samples, and take those raw videos' frames as positive samples. Besides, we bypass our timing verification to eliminate the mutual effect between these two verification algorithms. The experimental results are listed in the Table III, which shows a zero false positive error with 99.2% of accuracy rate.



(a) scenario



(b) results

Fig. 13: Primitive attack.

TABLE II: Four different screens.

	Screen	Resolution	Pixel Density
1	HUAWEI P10	1920*1080	432(ppi)
2	iPhone SE	1136*640	326(ppi)
3	AOC Monitor (e2450Swh)	1920*1080	93(ppi)
4	EIZO Monitor (ev2455)	1920*1200	95(ppi)

When applied with the testing data set, the accuracy is 98.8%. There are only 75 frames are incorrectly labeled, with all the negative samples labeled correctly. After analyzing these 75 frames, we found it may result from three reasons:

- Illumination. When the distance between face and screen is far and the environmental illumination is high, the captured response will be too obscure to be labeled correctly.
- Saturation. Due to device limitations, video frames taken in dark scenarios, will have many saturated pixels, even having adjusted the sensitivity of optical sensors. As described in Section IV-B1, it is necessary to remove these saturated pixels to satisfy the formulas.

TABLE III: Experimental results of face verification.

	Training Ps	Training Ns	Testing Ps	Testing Ns
Total	20931	20931	3000	3000
Incorrect	329	0	75	0

- Vibration. Drastic head shaking and intensive vibration also fades our performance. Especially, we will not do so well on frames at the beginning and end of the captured video.

The above results showed that we can detect all the attacks with a small false negative error, which provides another security guarantee besides the response timing mentioned above.

D. Evaluation on Robustness

There are mainly two elements that could affect the performance of our proposed method: illumination and vibration. We have carefully designed six scenarios to further investigate their impacts.

- scenario 1: We instruct participants to stand in a continuous lighting room as motionless as possible. And the button was hidden during the first 15 seconds to let participants produce a long video clip.
- scenario 2: We instruct participants to take a subway train. The vibration is intermittent and lighting condition is changing all the time.
- scenario 3: We instruct participants to walk on our campus as they usually do during a sunny day.
- scenario 4: We instruct participants to hover under penthouses during a cloudy day.
- scenario 5: We instruct participants to walk downstairs at their usual speed in rooms.
- scenario 6: We instruct participants to walk down a slope outside during nights.

We summarize the features of these scenarios in Table IV.

The results are shown on Fig 14. In ideal environments (scenario 1), our method is perfect and the accuracy is high as 99.83%. In normal cases (scenario 4), our method is also excellent with the 99.17% accuracy. And the sunlight (scenario 3) causes ignorable effects on the result, as long as the frontal camera does not face the sun directly. Comparing scenario 5 with 3, we infer the vibration causes more effect than the sunlight. Besides, dark is a devil (scenario 6) which reduces the accuracy to 97.33%, the lowest one. In our method, we cannot use the function of auto white balance (AWB) embedded in our devices, due to the fundamental requirement of our method. Adjusting the sensitivity of sensors, we just can limitedly reduce the effect of saturation, while keeping enough effectiveness. Limited by this constraint, the result is acceptable. For the complex case (scenario 2), the accuracy, 97.83%, is not bad. In this scenario, our device is being tested by many factors including unpredictable impacts, glare lamps and quickly changed shadows.

To further explore the impacts caused by vibrations, we built another experiment where we leveraged the six parameters generated by face tracking algorithm, and assembled them as a single value, ν, to measure the intensity of vibration. The details are illustrated in Algorithm 2, where $\{T_j\}$ is the sequence of the transformation matrix (Section IV-B2) and N is the number of frames.

TABLE IV: Features of scenes.

	Scenario 1	Scenario 2	Scenario 3	Scenario 4	Scenario 5	Scenario 6
Illumination	good	varying	intense	normal	normal	dark
Vibration	no	intermittent	normal	normal	intense	intense

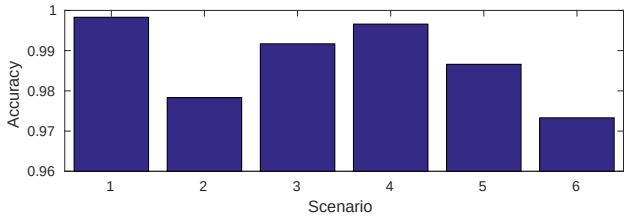
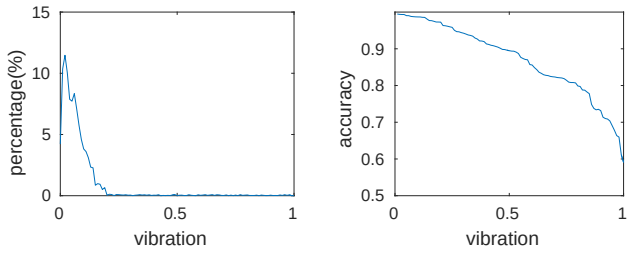


Fig. 14: Performance on different scenarios.

Algorithm 2 Algorithm to measure intensity of vibration.

INPUT: $\{T_j\}, N$
OUTPUT: ν
1: **for** $j = 1 \ \ to \ \ N$ **do**
2: Extract face shifting, $(\alpha_j, \beta_j, \gamma_j)$
3: Extract face rotation, $(\iota_j, \zeta_j, \eta_j)$
4: **end for**
5: Calculate mean values: $\bar{\alpha}, \bar{\beta}, \bar{\gamma}, \bar{\iota}, \bar{\zeta}$ and $\bar{\eta}$
6: **for** $i = j \ \ to \ \ N$ **do**
7: $\mu_j = \frac{\alpha_j}{\bar{\alpha}} + \frac{\beta_j}{\bar{\beta}} + \frac{\gamma_j}{\bar{\gamma}} + \frac{\iota_j}{\bar{\iota}} + \frac{\zeta_j}{\bar{\zeta}} + \frac{\eta_j}{\bar{\eta}}$
8: **end for**
9: $\nu = std(\{\mu_j\})$

Fig 15a shows the distribution of intensity. And Fig 15b shows the relation between vibration intensity and accuracy. We divided all the intensity by the maximum value. From both figures, we can infer that vibration will produce side effects to our method and the most drastic vibration will reduce the accuracy to 60%. But, in general cases where the vibration is not that big, our method can perform very well. This means our method indeed is robust under normal vibration conditions. Particularly, when the intensity reaches 0.5, we still hold 89% accuracy.



(a) distribution (b) relation

Fig. 15: Vibration effect.

In conclusion, our good robustness to vibration and illumination provides a good reliability and user experience. Besides, it excludes a potential attack scenario where adversary naively increases the vibration density.

E. Computational and Storage Cost

The time costs of our method depend on concrete devices. If we run our method in the back-end server (say a laptop), the time needed to deal with 300 frames is less than 1 seconds, and the difference among time costs of our 3 steps is subtle. Here, we amplify this difference by running our method on a smartphone (Nexus 6) with a single core, and the resolution of all the frames are kept on 1920*1080. The time costs are shown in Table V.

TABLE V: Time cost of implementation on smartphone.

Number of frames	50	100	200	300
Face extraction	6.11	11.70	20.12	28.63
Timing verification	0.03	0.05	0.11	0.22
Face verification	0.08	0.12	0.20	0.27
Total	6.22 (s)	11.87 (s)	20.43 (s)	29.12 (s)

We discover that the most time-consuming step is the face extraction, which depends on algorithms we choose and the precision of face detection we want to achieve. The lower precision, the lower resolution of the input frame is needed and thus less time is needed. Particularly, if we shrink the input frame to half its size, the time will be reduced to about 1 second to extract the faces on 50 frames. The other way to reduce the time cost is leveraging the back-end server (the Cloud) in parallel, as we mentioned above. In practice, we keep the camera continuously recording the user's video and, parallely, sent ".mp4" files with each containing 30 frames recorded in one second, to our server through a 4G network (with about 12 Mbps of bandwidth in our experiment) for every second. Transferring one that file will consume 1.1 MB bandwidth. Once receiving a video, the server will perform our verifications on it and judge whether the user is benign. If the result of any second is negative, we regard this whole authentication session as a malicious attempt. Table VI demonstrates the time cost of this process. Compared with the implementation only on the smartphone, using cloud can significantly reduces the waiting and thus greatly improved user experiences.

TABLE VI: Time cost of implementation using cloud.

Number of frames	50	100	200	300
Recording in Front	1.67	3.34	6.67	10
Verifying in Cloud	2.22	3.62	7.21	10.82
Time to Wait	0.55 (s)	0.28 (s)	0.54 (s)	0.82 (s)

The storage space we need is the same as the size of captured videos, and the storage complexity is $O(NM)$. In real tests, 8.3Mb memory space is enough to store a video consisting 100 frames in JPG format.

VII. Related Works

Various liveness detection techniques have been proposed in the past decades. In this section, we discuss differences between our method and those most relevant previous studies.

Our method could be categorized as a texture extraction method, according to the classification in Chakraborty's survey [5]. The traditional methods in this category mainly

use various *descriptors* to extract features of images and pass features through a classifier to obtain the final result. For instance, Arashloo et al. [2] used multi-scale dynamic binarized statical features; Benlamoudi et al. [2] used active shape models with steam and LBP; Wen et al. [25] analyzed distortion using 4 different features, etc. These methods work well under experimental conditions. But in our adversary model, the attacker can forge a perfect face that would defeat their approaches. In contrast, our method checks the geometric shape of the subject under authentication, and detect whether there are abnormal delays between responses and challenges. Even the adversary is technically capable of creating a perfect forged response, the time required in doing so will fail them. Besides, previous works may fail due to the sub-optimal environmental conditions. However, our method is robust to that, as demonstrated in the evaluation part.

Our method is also a challenge-response protocol. The traditional protocols are based on human reactions. Comparing to them, our responses can be generated at the speed of light. Li et al. [17] proposed a new protocol that records the inertial sensors' data while the user is moving around the mobile phone. If the data is consistent with the video captured by the mobile phone, the user is judged as a legitimate one. This method's challenge is the movement of mobile phone which is controlled by the user and measured by sensors. And the response is user's facial video which is also produced by the user. This method's security guarantee is based on the precise estimation of head poses. But we argued that the accuracy cannot be high enough in wild environment for two reasons: first, as mentioned by the authors, the estimation algorithm has about 7 degrees deviation; second, hand trembling produces side effect to the precision of the mobile sensors. In contrast, our approach is more robust, because, firstly, the challenges are fully out of attackers' control, and, secondly, our security guarantees are based on detecting the indelible delay, rather than the accurate estimation of the unstable head position.

Besides above methods, there is a close work published by Kim et al. [12] who found that the diffusion speed is the distinguishing characteristic between real faces and fake faces. The reason is that the real face has more stereo shape which makes the reflection random and irregular. But this passive method will not work, when the environmental light is inefficient. From the figures shown in their paper, we can hardly distinguish the so-called binarized reflectance maps of malicious responses from legitimate responses, and these "vague" maps are fed to SVM for the final decision. So we argue that this approach cannot defeat such attackers who have the ability to forge a perfect fake face. In contrast, our security guarantee is not only based on the stereo shape, but also the delay between responses and challenges. It's a very high bar for adversaries to forge a perfect response in such critical time. Another method leveraging reflection is proposed by Rudd et al. [20]. The authors added two different polarization devices on the camera. And these devices impede the most of incoming light except the light in the particular direction. Comparing to this approach, our method does not require special devices and is more practical to use.

Specially, our work has overlaps with Andrew Bud's patent [4] using also the light reflection to do the authentication. However, we focus on the security instead of functionality as they have done. Without our timing verification, as we demonstrated, there is no security guarantee and it is weak to defend against MFF attacks. And our work is independent from theirs.

In general, compared with above relative works, Face Flashing is an active and effective approach with strong security guarantee on time.

VIII. Discussion

Resilience to novel attacks. An attack proposed by Mahmood et al. [21] demonstrated that an attacker could impersonate the victim by placing a customized mask around his eyes. Although such an attack can deceive the state-of-the-art face recognition system, however, we believe it will be defeated by our method, as paper masks around the eyes can be easily detected by our neural network model in the verification of face (see Fig 8a and 8c).

Challenge colors. We used 8 different colors in our experiments. Considering the length of our challenge sequence, we believe these 8 colors are enough to provide a strong security guarantee. Because our security guarantee is achieved by detecting the *delays*. If the adversary falsely infers one challenge, the *delay* will be detected and her attempt will fail. Of course, we can easily increase the space of the challenge sequences by using the striped pictures with a more sophisticated algorithm.

Authentication time. Our method needs a few seconds to gather enough responses for authentication. As we mentioned in data collection's part, 3 seconds is a reasonable default setting. In this period, we can choose sufficient responses with high quality, and the user can complete the instructed expression. Essentially, 1 second is enough for our method to finish the work, but the user will be in a hurry.

Other applications of our techniques. One interesting application of our method is to improve the accuracy of state-of-the-art face recognition algorithms by distilling the personal information contained in the geometric shape. We believe the shape is unique. The combined method will have stronger ability to prevent advanced future attacks.

IX. Limitations

The silicone mask may pass our system. But, this mask is hard to be fabricated (3D printed) due to the reasons mentioned in Section II-B. And our system has the potential to defeat it completely, owing to our unique challenges: lights of different wavelength (colors). According to previous studies [26], light reflected from human skin has an "albedo curve", the curve depicting reflectance of different wavelengths. Therefore, the reflections from different surfaces can be distinguished by discernible albedo curves, which enables Face Flashing to recognize attackers wearing such "soft" masks. However, this technique is sophisticated and deserves another paper.

Even though we raise the bar of the attacks, we cannot totally neutralize adversaries' advantages coming from super devices. They still have a chance to pass our system, if they somehow use an ultrahigh-speed camera (FASTCAM SA1.1

with 675000fps), an ultrahigh-speed screen in the similar level (says with 100000Hz), and the solution to reduce the transmission and buffering delays. In this situation, adversaries can instantly forge the response to every challenge with small delays and subtle variance, so our protocol will fail. However, this attack is expensive and sophisticated. On the other side, we can mitigate this threat, to some extent, by flashing more finely striped challenges (or chessboard-like patterns), but, with better screen and camera.

X. Conclusion

In this paper, we proposed a novel challenge-response protocol, Face Flashing, to defeat the main threats against face authentication system—the *2D dynamic* attacks. We have systematically analyzed our method and illustrated that our method has strong security guarantees. We implemented a prototype that does verifications both on time and the face. We have demonstrated that our method has high accuracy in various environments and is robust to vibration and illumination. Experimental results prove that our protocol is effective and efficient.

Acknowledgment

We thank our shepherd Muhammad Naveed for his patient guidance on improving this paper, and anonymous reviewers for their insightful comments. We also want to thank Tao Mo and Shizhan Zhu for their supports on the face alignment and tracking algorithms. This work was partially supported by National Natural Science Foundation of China (NSFC) under Grant No. 61572415, Hong Kong S.A.R. Research Grants Council (RGC) Early Career Scheme/General Research Fund No. 24207815 and 14217816.

References

[1] W. Bao, H. Li, N. Li, and W. Jiang, "A liveness detection method for face recognition based on optical flow field," in *Image Analysis and Signal Processing, 2009. IASP 2009. International Conference on*. IEEE, 2009, pp. 233–236.

[2] A. Benlamoudi, D. Samai, A. Ouafi, A. Taleb-Ahmed, S. E. Bekhouche, and A. Hadid, "Face spoofing detection from single images using active shape models with stasm and lbp," in *Proceeding of the Troisime CONFERENCE INTERNATIONALE SUR LA VISION ARTIFICIELLE CVA 2015*, 2015.

[3] D. H. Brainard and B. A. Wandell, "Analysis of the retinex theory of color vision," *JOSA A*, vol. 3, no. 10, pp. 1651–1661, 1986.

[4] A. Bud, "Online pseudonym verification and identity validation," U.S. Patent 9 479 500B2, Oct 25, 2016.

[5] S. Chakraborty and D. Das, "An overview of face liveness detection," *International Journal on Information Theory*, vol. 3, no. 2, 2014.

[6] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[7] W. Fernando, L. Udawatta, and P. Pathirana, "Identification of moving obstacles with pyramidal lucas kanade optical flow and k means clustering," in *2007 Third International Conference on Information and Automation for Sustainability*. IEEE, 2007, pp. 111–117.

[8] R. D. Findling and R. Mayrhofer, "Towards face unlock: on the difficulty of reliably detecting faces on mobile phones," in *Proceedings of the 10th International Conference on Advances in Mobile Computing & Multimedia*. ACM, 2012, pp. 275–280.

[9] H.-K. Jee, S.-U. Jung, and J.-H. Yoo, "Liveness detection for embedded face recognition system," *International Journal of Biological and Medical Sciences*, vol. 1, no. 4, pp. 235–238, 2006.

[10] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*. ACM, 2014, pp. 675–678.

[11] G. Kim, S. Eum, J. K. Suhr, D. I. Kim, K. R. Park, and J. Kim, "Face liveness detection based on texture and frequency analyses," in *2012 5th IAPR International Conference on Biometrics (ICB)*. IEEE, 2012, pp. 67–72.

[12] W. Kim, S. Suh, and J.-J. Han, "Face liveness detection from a single image via diffusion speed model," *IEEE Transactions on Image Processing*, vol. 24, no. 8, pp. 2456–2465, 2015.

[13] K. Kollreider, H. Fronthaler, and J. Bigun, "Non-intrusive liveness detection by face images," *Image and Vision Computing*, vol. 27, no. 3, pp. 233–244, 2009.

[14] K. Kollreider, H. Fronthaler, M. I. Faraj, and J. Bigun, "Real-time face detection and motion analysis with application in liveness assessment," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 3, pp. 548–558, 2007.

[15] A. Lagorio, M. Tistarelli, M. Cadoni, C. Fookes, and S. Sridharan, "Liveness detection based on 3d face shape analysis," in *Biometrics and Forensics (IWBF), 2013 International Workshop on*. IEEE, 2013, pp. 1–4.

[16] J. Li, Y. Wang, T. Tan, and A. K. Jain, "Live face detection based on the analysis of fourier spectra," in *Defense and Security*. International Society for Optics and Photonics, 2004, pp. 296–303.

[17] Y. Li, Y. Li, Q. Yan, H. Kong, and R. H. Deng, "Seeing your face is not enough: An inertial sensor-based liveness detection for face authentication," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2015, pp. 1558–1569.

[18] J. Määttä, A. Hadid, and M. Pietikainen, "Face spoofing detection from single images using micro-texture analysis," in *Biometrics (IJCB), 2011 international joint conference on*. IEEE, 2011, pp. 1–7.

[19] R. Raghavendra and C. Busch, "Robust 2d/3d face mask presentation attack detection scheme by exploring multiple features and comparison score level fusion," in *Information Fusion (FUSION), 2014 17th International Conference on*. IEEE, 2014, pp. 1–7.

[20] E. M. Rudd, M. Gunther, and T. E. Boult, "Paraph: presentation attack rejection by analyzing polarization hypotheses," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2016, pp. 103–110.

[21] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proceedings of the 23rd ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[22] R. S. Smith and T. Windeatt, "Facial expression detection using filtered local binary pattern features with ecoc classifiers and platt scaling." in *WAPA*, 2010, pp. 111–118.

[23] L. Sun, G. Pan, Z. Wu, and S. Lao, "Blinking-based live face detection using conditional random fields," in *International Conference on Biometrics*. Springer, 2007, pp. 252–260.

[24] T. Wang, J. Yang, Z. Lei, S. Liao, and S. Z. Li, "Face liveness detection using 3d structure recovered from a single camera," in *Biometrics (ICB), 2013 International Conference on*. IEEE, 2013, pp. 1–6.

[25] D. Wen, H. Han, and A. K. Jain, "Face spoof detection with image distortion analysis," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 4, pp. 746–761, 2015.

[26] T. Weyrich, W. Matusik, H. Pfister, J. Lee, A. Ngan, H. Wann, and M. G. Jensen, "A measurement-based skin reflectance model for face," 2005.

[27] Y. Xu, T. Price, J.-M. Frahm, and F. Monrose, "Virtual u: Defeating face liveness detection by building virtual models from your public photos," in *25th USENIX Security Symposium (USENIX Security 16)*. Austin, TX: USENIX Association, Aug. 2016, pp. 497–512.

[28] B. Yang, J. Yan, Z. Lei, and S. Z. Li, "Convolutional channel features," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 82–90.

[29] S. Zhu, C. Li, C. C. Loy, and X. Tang, "Unconstrained face alignment via cascaded compositional learning," 2016.